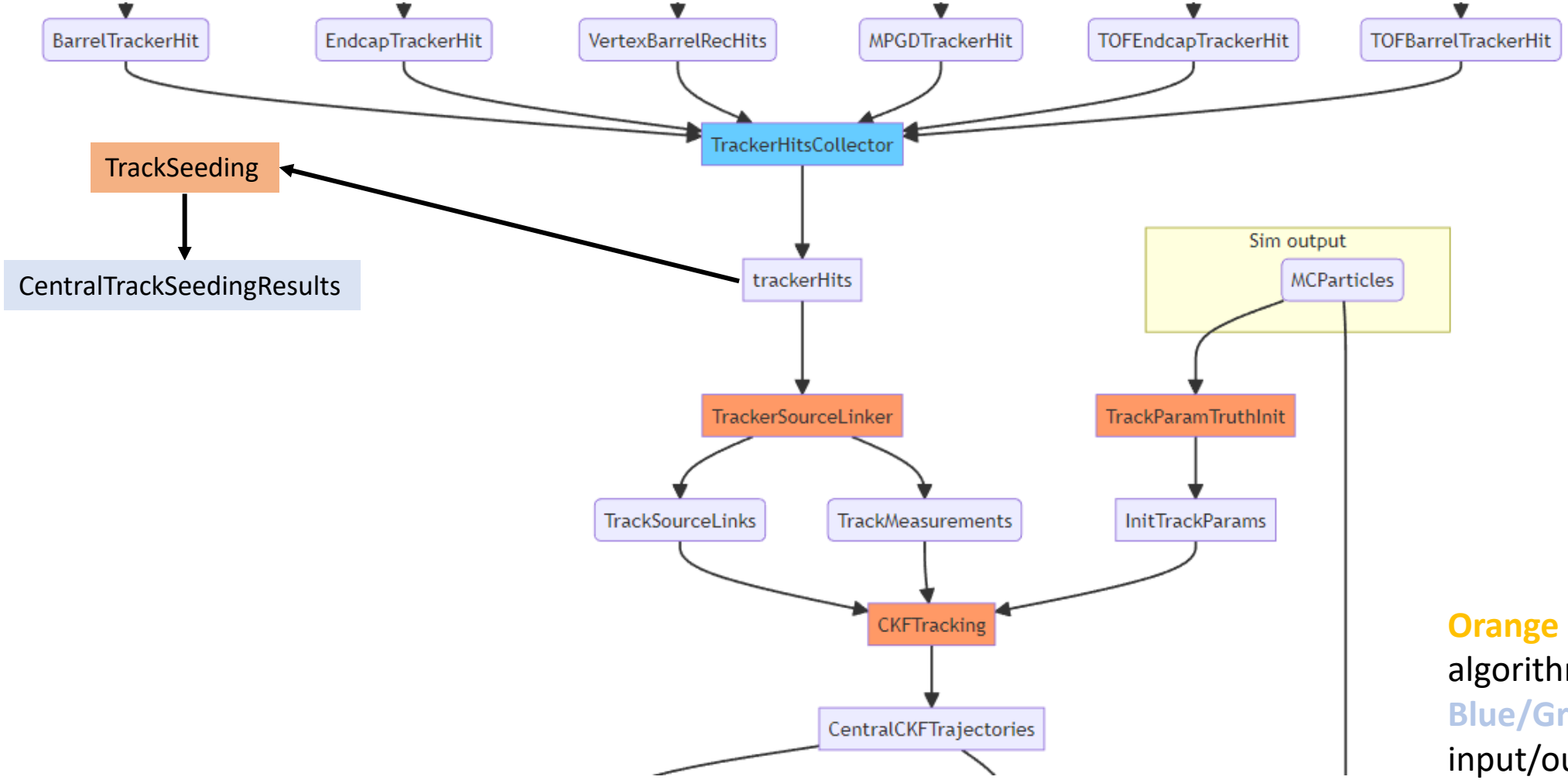
Using realistic seeding for track fitting in EICRecon

Barak Schmookler

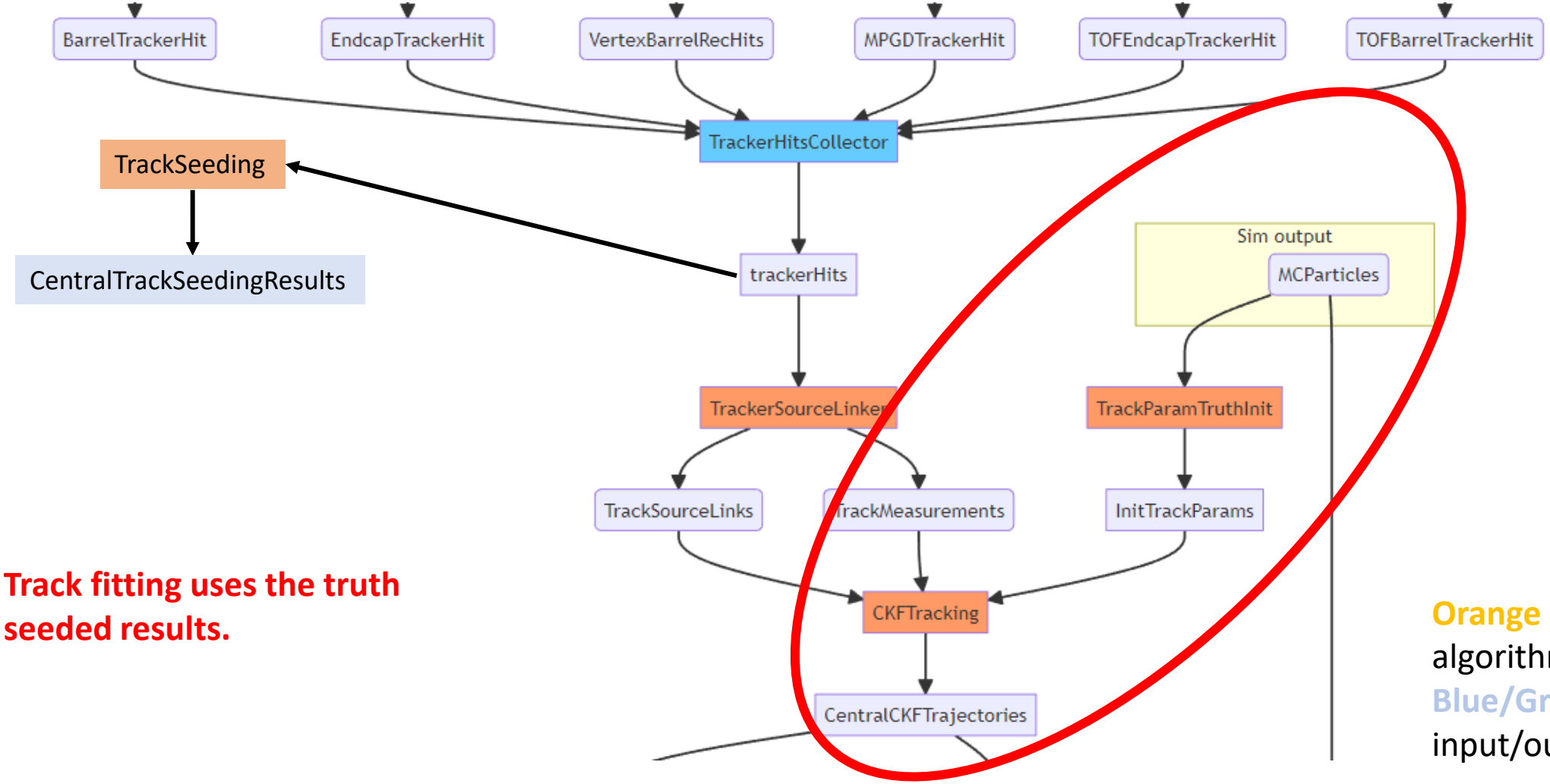
Tracking logic in EICRecon



Orange boxes are algorithms.
Blue/Grey boxes are input/output data.

https://eic.github.io/EICrecon/#/design/tracking

Tracking logic in EICRecon

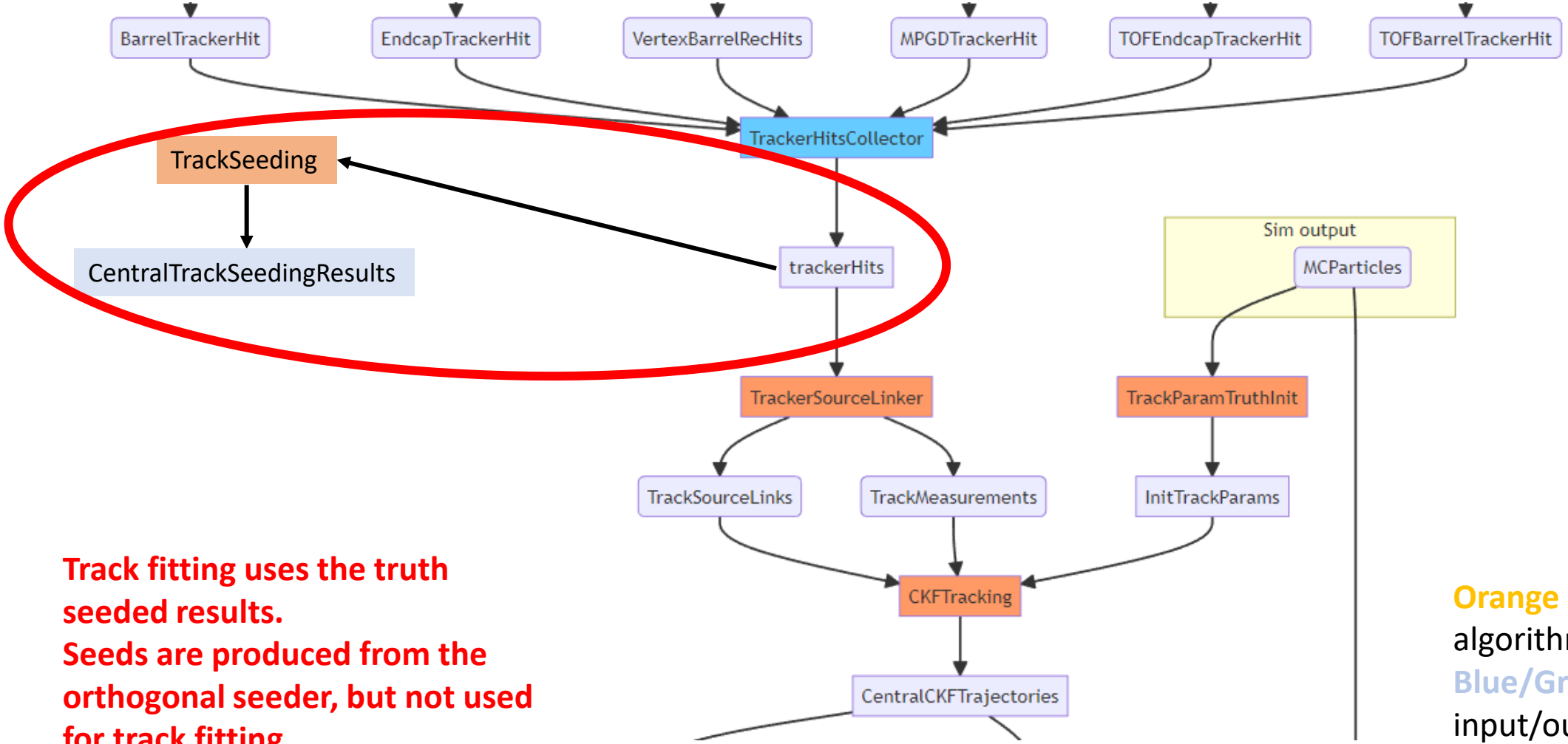


Track fitting uses the truth seeded results.

Orange boxes are algorithms.
Blue/Grey boxes are input/output data.

https://eic.github.io/EICrecon/#/design/tracking

Tracking logic in EICRecon

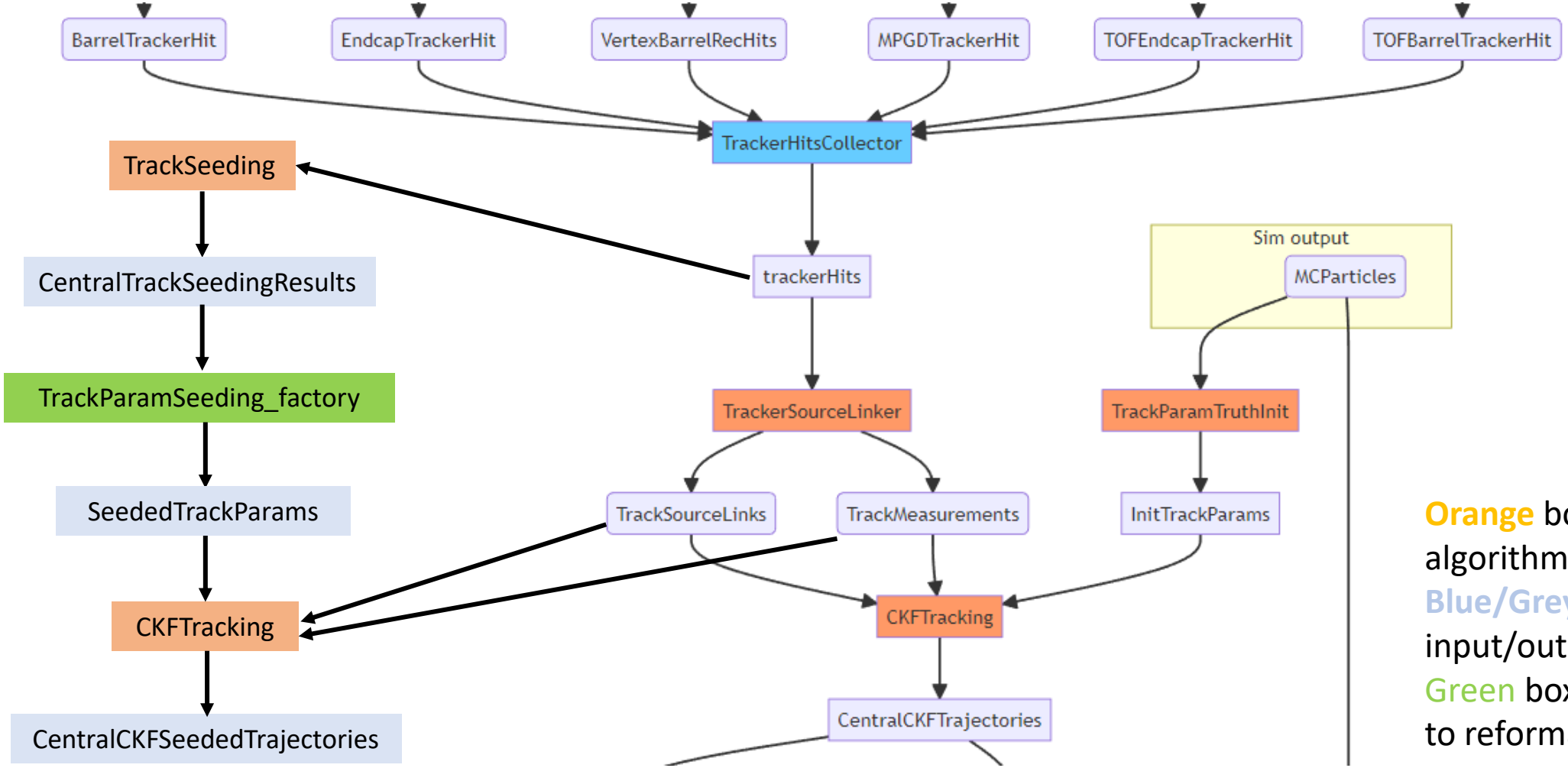


Track fitting uses the truth seeded results.
Seeds are produced from the orthogonal seeder, but not used for track fitting.

Orange boxes are algorithms.
Blue/Grey boxes are input/output data.

https://eic.github.io/EICrecon/#/design/tracking

Update to use real seed for tracking



Orange boxes are algorithms.
Blue/Grey boxes are input/output data.
Green box is a factory to reformat data type.

https://eic.github.io/EICrecon/#/design/tracking

How this can be used

- The code lives in our track-QA branch:

  https://github.com/eic/EICrecon/tree/track-qa-barak

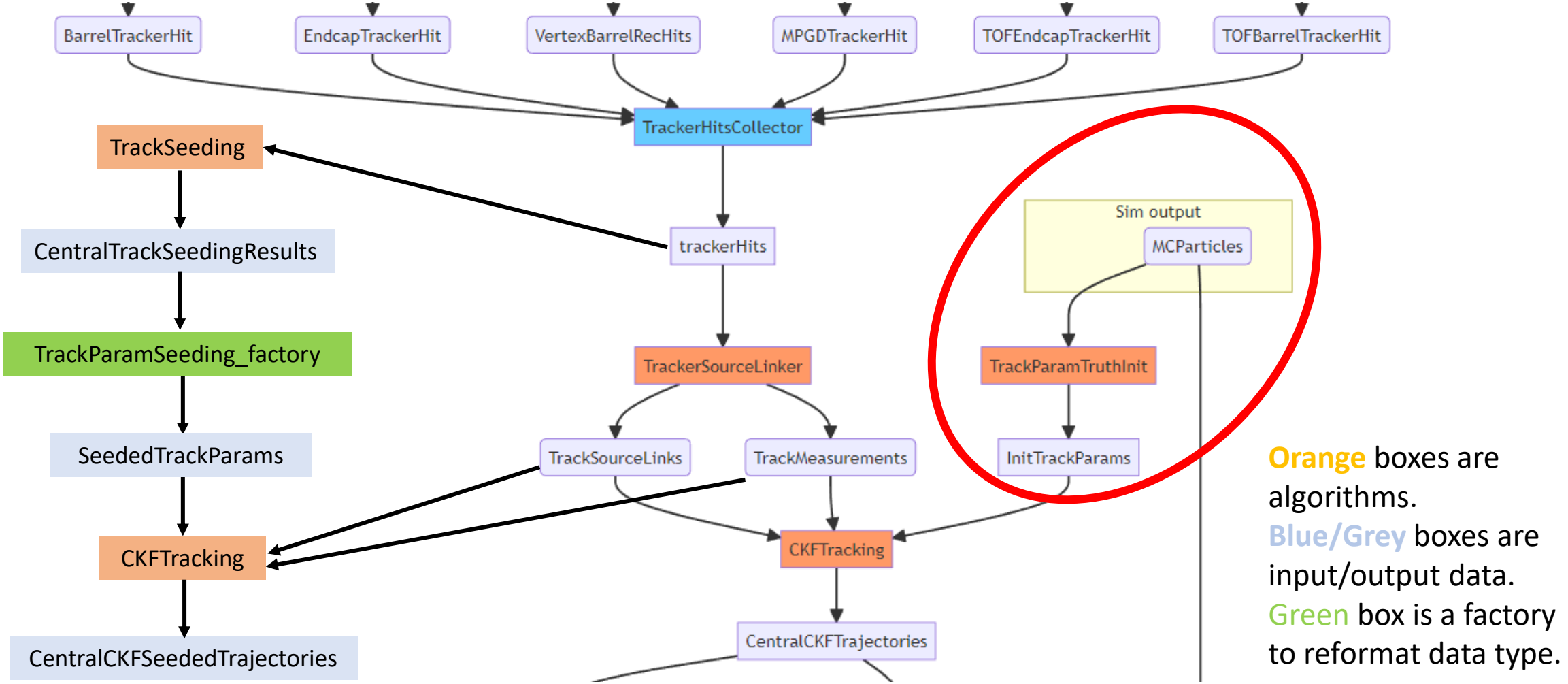
- The new datatypes and factories shown above do not affect the previous workflow. So, they can be merged into the main branch without causing any changes to the standard output ROOT file.
- A user can access the tracks which use the realistic seeding by using the following in a Plugin:

```
auto trajectories = event->Get<eicrecon::TrackingResultTrajectory>("CentralCKFSeededTrajectories");
```

instead of (for truth seeded tracks):

```
auto trajectories = event->Get<eicrecon::TrackingResultTrajectory>("CentralCKFTrajectories");
```

Investigation into the seeding parameters: truth seeding



https://eic.github.io/EICrecon/#/design/tracking

Truth seeded parameters

```
81      // build some track cov matrix
82      Acts::BoundSymMatrix cov                    = Acts::BoundSymMatrix::Zero();
83      cov(Acts::eBoundLoc0, Acts::eBoundLoc0)     = 1000*um*1000*um;
84      cov(Acts::eBoundLoc1, Acts::eBoundLoc1)     = 1000*um*1000*um;
85      cov(Acts::eBoundPhi, Acts::eBoundPhi)       = 0.05*0.05;
86      cov(Acts::eBoundTheta, Acts::eBoundTheta)   = 0.01*0.01;
87      cov(Acts::eBoundQOverP, Acts::eBoundQOverP) = (0.1*0.1) / (GeV*GeV);
88      cov(Acts::eBoundTime, Acts::eBoundTime)     = 10.0e9*ns*10.0e9*ns;
89
90      Acts::BoundVector  params;
91      params(Acts::eBoundLoc0)   = 0.0 * mm ;  // cylinder radius
92      params(Acts::eBoundLoc1)   = 0.0 * mm ;  // cylinder length
93      params(Acts::eBoundPhi)    = phi;
94      params(Acts::eBoundTheta)  = theta;
95      params(Acts::eBoundQOverP) = charge / (pinit * GeV);
96      params(Acts::eBoundTime)   = part->getTime() * ns;
97
98      //// Construct a perigee surface as the target surface
99      auto pSurface = Acts::Surface::makeShared<Acts::PerigeeSurface>(
100             Acts::Vector3{part->getVertex().x * mm, part->getVertex().y * mm, part->getVertex().z * mm});
101
102     //params(Acts::eBoundQOverP) = charge/p;
103     auto result = new eicrecon::TrackParameters({pSurface, params, charge, cov});
104     return result;
```

This is what gets passed to the tracking algorithm.

These parameters and surface come from the generated particle.

The covariance matrix of the seed is hardcoded.

https://github.com/eic/EICrecon/blob/main/src/algorithms/tracking/TrackParamTruthInit.cc

Cuts applied on truth seeds

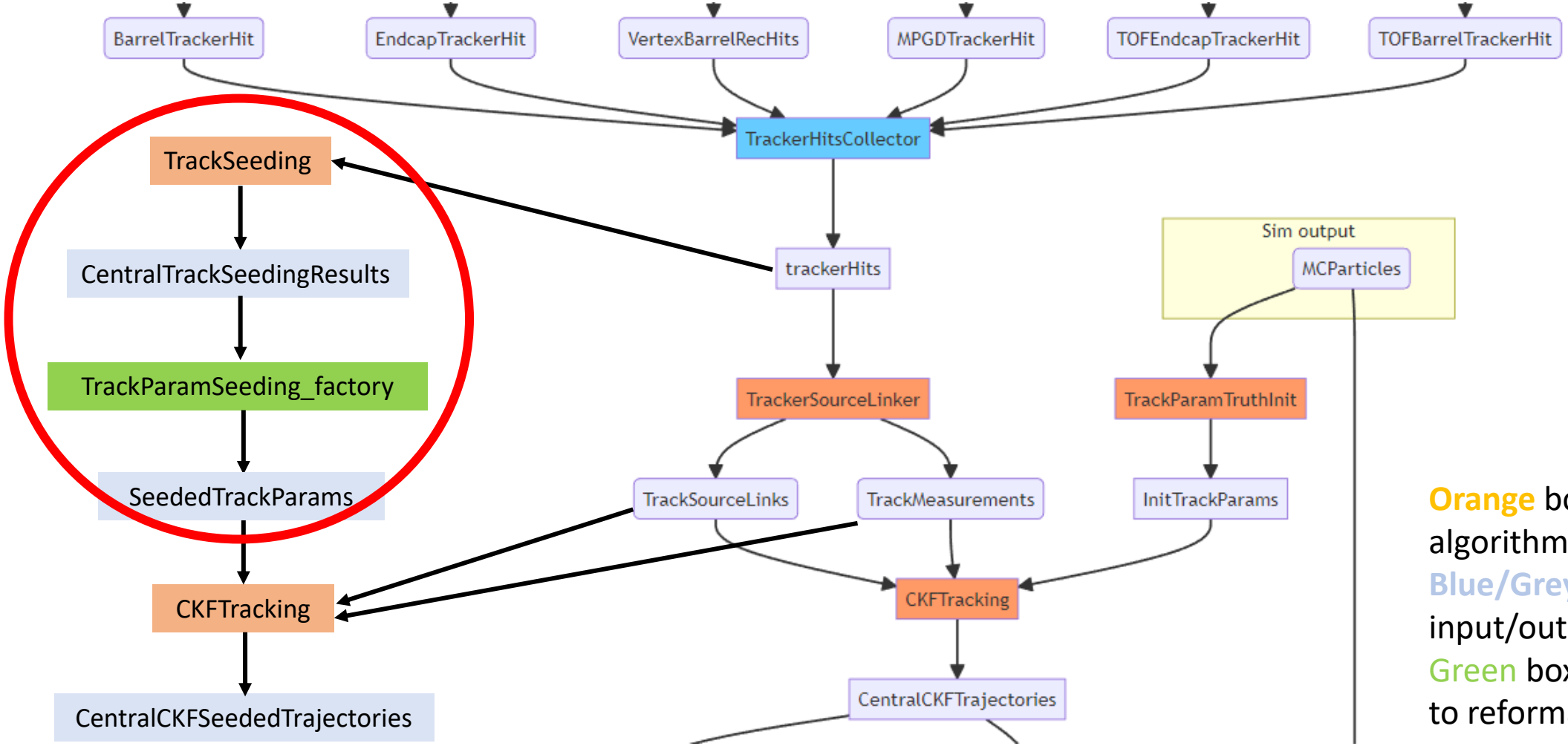
```
struct TrackParamTruthInitConfig {

    double m_maxVertexX       = 80  * Acts::UnitConstants::mm;
    double m_maxVertexY       = 80  * Acts::UnitConstants::mm;
    double m_maxVertexZ       = 200 * Acts::UnitConstants::mm;
    double m_minMomentum      = 100 * Acts::UnitConstants::MeV;
    double m_maxEtaForward    = 4.0;
    double m_maxEtaBackward   = 4.1;
    double m_momentumSplit    = 0.0;
    double m_momentumSmear    = 0.0;

};
```

https://github.com/eic/EICrecon/blob/main/src/algorithms/tracking/TrackParamTruthInitConfig.h

Investigation into the seeding parameters: orthogonal seeding



Orange boxes are algorithms.
Blue/Grey boxes are input/output data.
Green box is a factory to reformat data type.

https://eic.github.io/EICrecon/#/design/tracking

Investigation into the seeding parameters: orthogonal seeding

Seeder writes out an edm4eic::TrackParameters data type which can be accessed in ROOT file

```
115         edm4eic::TrackParameters *params = new edm4eic::TrackParameters{
116           -1, // type --> seed(-1)
117           {(float)localpos(0), (float)localpos(1)}, // 2d location on surface
118           {0.1,0.1}, //covariance of location
119           theta, //theta rad
120           atan2(xyHitPositions.at(0).second, xyHitPositions.at(0).first), // phi of first hit (rad)
121           qOverP, // Q/p [e/GeV]
122           {0.05,0.05,0.05}, // covariance on theta/phi/q/p
123           10, // time in ns
124           0.1, // error on time
125           (float)charge // charge
126
127         };
128
129         trackparams.push_back(params);
130       }
```

https://github.com/eic/EICrecon/blob/main/src/algorithms/tracking/TrackSeeding.cc

Convert to eicrecon::TrackParameters data type as input for track fitting

```
55            //Seed Parameters
56            Acts::BoundVector  params;
57            params(Acts::eBoundLoc0)   = (*aseed)->getLoc().a * mm ;  // cylinder radius
58            params(Acts::eBoundLoc1)   = (*aseed)->getLoc().b * mm ;  // cylinder length
59            params(Acts::eBoundPhi)    = (*aseed)->getPhi();
60            params(Acts::eBoundTheta)  = (*aseed)->getTheta();
61            params(Acts::eBoundQOverP) = (*aseed)->getQOverP()/ GeV;
62            params(Acts::eBoundTime)   = (*aseed)->getTime() * ns;
63
64            //Get charge
65            double charge = (*aseed)->getCharge();
66
67            //Build seed Covariance matrix
68            Acts::BoundSymMatrix cov                    = Acts::BoundSymMatrix::Zero();
69            cov(Acts::eBoundLoc0, Acts::eBoundLoc0)     = std::pow( (*aseed)->getLocError().xx ,2)*mm*mm;
70            cov(Acts::eBoundLoc1, Acts::eBoundLoc1)     = std::pow( (*aseed)->getLocError().yy,2)*mm*mm;
71            cov(Acts::eBoundPhi, Acts::eBoundPhi)       = std::pow( (*aseed)->getMomentumError().xx,2);
72            cov(Acts::eBoundTheta, Acts::eBoundTheta)   = std::pow( (*aseed)->getMomentumError().yy,2);
73            cov(Acts::eBoundQOverP, Acts::eBoundQOverP) = std::pow( (*aseed)->getMomentumError().zz,2) / (GeV*GeV);
74            cov(Acts::eBoundTime, Acts::eBoundTime)     = std::pow( (*aseed)->getTimeError(),2)*ns*ns;
75
76            //Construct a perigee surface as the target surface
77            auto pSurface = Acts::Surface::makeShared<Acts::PerigeeSurface>(Acts::Vector3(0,0,0));
```

https://github.com/eic/EICrecon/blob/track-qa-barak/src/global/tracking/TrackParamSeeding_factory.cc

Investigation into the seeding parameters: orthogonal seeding

Seeder writes out an edm4eic::TrackParameters data type which can be accessed in ROOT file

```
115         edm4eic::TrackParameters *params = new edm4eic::TrackParameters{
116           -1, // type --> seed(-1)
117           {(float)localpos(0), (float)localpos(1)}, // 2d location on surface
118           {0.1,0.1}, //covariance of location
119           theta, //theta rad
120           atan2(xyHitPositions.at(0).second, xyHitPositions.at(0).first), // phi of first hit (rad)
121           qOverP, // Q/p [e/GeV]
122           {0.05,0.05,0.05}, // covariance on theta/phi/q/p
123           10, // time in ns
124           0.1, // error on time
125           (float)charge // charge
126
127         };
128
129         trackparams.push_back(params);
130       }
```

The covariance matrix of the seed is also hardcoded here.

https://github.com/eic/EICrecon/blob/main/src/algorithms/tracking/TrackSeeding.cc

Convert to eicrecon::TrackParameters data type as input for track fitting

```
55          //Seed Parameters
56          Acts::BoundVector  params;
57          params(Acts::eBoundLoc0)   = (*aseed)->getLoc().a * mm ;  // cylinder radius
58          params(Acts::eBoundLoc1)   = (*aseed)->getLoc().b * mm ;  // cylinder length
59          params(Acts::eBoundPhi)    = (*aseed)->getPhi();
60          params(Acts::eBoundTheta)  = (*aseed)->getTheta();
61          params(Acts::eBoundQOverP) = (*aseed)->getQOverP()/ GeV;
62          params(Acts::eBoundTime)   = (*aseed)->getTime() * ns;
63
64          //Get charge
65          double charge = (*aseed)->getCharge();
66
67          //Build seed Covariance matrix
68          Acts::BoundSymMatrix cov                    = Acts::BoundSymMatrix::Zero();
69          cov(Acts::eBoundLoc0, Acts::eBoundLoc0)     = std::pow( (*aseed)->getLocError().xx ,2)*mm*mm;
70          cov(Acts::eBoundLoc1, Acts::eBoundLoc1)     = std::pow( (*aseed)->getLocError().yy,2)*mm*mm;
71          cov(Acts::eBoundPhi, Acts::eBoundPhi)       = std::pow( (*aseed)->getMomentumError().xx,2);
72          cov(Acts::eBoundTheta, Acts::eBoundTheta)   = std::pow( (*aseed)->getMomentumError().yy,2);
73          cov(Acts::eBoundQOverP, Acts::eBoundQOverP) = std::pow( (*aseed)->getMomentumError().zz,2) / (GeV*GeV);
74          cov(Acts::eBoundTime, Acts::eBoundTime)     = std::pow( (*aseed)->getTimeError(),2)*ns*ns;
75
76          //Construct a perigee surface as the target surface
77          auto pSurface = Acts::Surface::makeShared<Acts::PerigeeSurface>(Acts::Vector3(0,0,0));
```

https://github.com/eic/EICrecon/blob/track-qa-barak/src/global/tracking/TrackParamSeeding_factory.cc

ACTS seeder in Juggler

```
308     // Set up the track parameters covariance (the same for all
309     // tracks)
310     m_covariance(Acts::eBoundLoc0, Acts::eBoundLoc0) =
311         std::pow(m_cfg.sigmaLoc0, 2);
312     m_covariance(Acts::eBoundLoc1, Acts::eBoundLoc1) =
313         std::pow(m_cfg.sigmaLoc1, 2);
314     m_covariance(Acts::eBoundPhi, Acts::eBoundPhi) =
315         std::pow(m_cfg.sigmaPhi, 2);
316     m_covariance(Acts::eBoundTheta, Acts::eBoundTheta) =
317         std::pow(m_cfg.sigmaTheta, 2);
318     m_covariance(Acts::eBoundQOverP, Acts::eBoundQOverP) =
319         std::pow(m_cfg.sigmaQOverP, 2);
320     m_covariance(Acts::eBoundTime, Acts::eBoundTime) =
321         std::pow(m_cfg.sigmaT0, 2);
322
323     return StatusCode::SUCCESS;
```

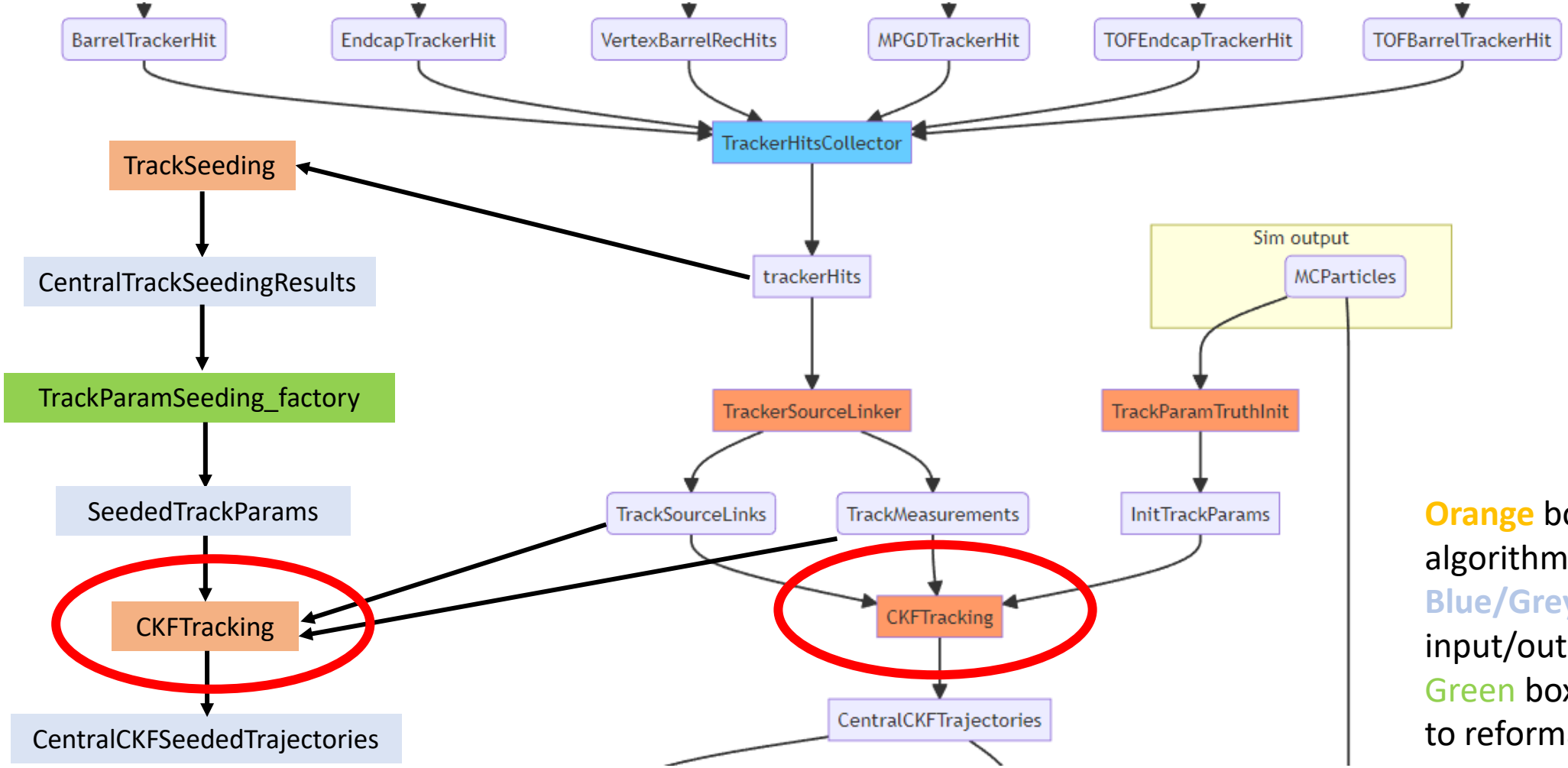
```
605                 initTrackParameters->emplace_back(
606                     surface->getSharedPtr(), params, charge,
607                     m_covariance);
```

The covariance matrix of the seed seems to also be set to a fixed value in the Juggler version.

The seeder requires the same four pieces of information as in the EICRecon. Both use the ACTS::SingleBoundTrackParameters class.

https://eicweb.phy.anl.gov/EIC/juggler/-/blob/main/JugTrack/src/components/TrackParamACTSSeeding.cpp

Tracking parameters



Orange boxes are algorithms.
Blue/Grey boxes are input/output data.
Green box is a factory to reformat data type.

https://eic.github.io/EICrecon/#/design/tracking

Tracking parameters

```
 9  namespace eicrecon {
10      struct CKFTrackingConfig {
11          std::vector<double> m_etaBins = {};  // {this, "etaBins", {}};
12          std::vector<double> m_chi2CutOff = {15.}; //{this, "chi2CutOff", {15.}};
13          std::vector<size_t> m_numMeasurementsCutOff = {10}; //{this, "numMeasurementsCutOff", {10}};
14      };
15  }
```

https://github.com/eic/EICrecon/blob/main/src/algorithms/tracking/CKFTrackingConfig.h

Summary

- We have added code in EICRecon which allows track fitting using the orthogonal seeder results.
- The output of this track fitting can be used in an EICRecon Plugin. The new code will have no effect on the normal output ROOT file which is created by EICRecon.
- Some investigation has been made into the details of the seeding parameters.